CLAIMS

1   1.   A method for performing coverability analysis
2 in software, comprising:
3   performing a static analysis of software under test
4 (SUT) so as to identify a plurality of dominating blocks
5 in the SUT;
6   formulating respective coverability tasks for the
7 dominating blocks of the SUT;
8   generating rules regarding behavior of the SUT
9 corresponding respectively to the coverability tasks;
10   for each of the rules, running a symbolic model
11 checker to test a behavioral model of the SUT, so as to
12 produce respective results for the rules; and
13   computing a coverability metric for the SUT
14 responsive to the results and the coverability tasks.

1   2.   A method according to claim 1, and comprising
2 writing the SUT in a programming language adapted to
3 define at least one of a group of elements comprising a
4 software element and a hardware element.

1   3.   A method according to claim 1, wherein
2 performing the static analysis of the SUT comprises:
3   identifying a set of dominating blocks in the SUT;
4 and
5   solving a subset cover problem on the set of
6 dominating blocks so as to identify the plurality of
7 dominating blocks.

1   4.   A method according to claim 3, wherein the set
2 of dominating blocks comprises a set of all dominating
3 blocks in the SUT, and wherein the plurality of
4 dominating blocks comprises fewer blocks than the set of
5 all dominating blocks in the SUT.

1   5.   A method according to claim 4, wherein running

'2 the symbolic model checker comprises performing a number

3   of executions of the symbolic model checker smaller than

4   a total number of all the dominating blocks in the SUT.

1   6.   A method according to claim 1, wherein

2   formulating the respective coverability tasks for the

3   dominating blocks of the SUT comprises formulating

4   coverability tasks by at least one of a group of methods

5   comprising manual formulation and automatic formulation.

1   7.   A method according to claim 1, wherein

2   generating the rules regarding behavior of the SUT

3   comprises generating rules by at least one of a group of

4   methods comprising manual generation and automatic

5   generation.

1   8.   A method according to claim 1, wherein running

2   the symbolic model checker to test the behavioral model

3   of the SUT comprises:

4   evaluating the respective results so as to determine

5   the truth or falsity of the rule; and

6   generating a list of uncoverable elements responsive

7   to the respective results.

1   9.   A method according to claim 1, wherein

2   generating the rules regarding behavior of the SUT

3   corresponding respectively to the coverability tasks

4   comprises instrumenting the SUT by adding one or more

5   statements and one or more auxiliary variables thereto,

6   so as to facilitate evaluation of the rules.

1   10. A method according to claim 9, wherein

2   instrumenting the SUT comprises:

3   determining a plurality of basic blocks comprised in

4   the SUT; and

5   for each basic block:

6   defining an auxiliary variable for the block;

7   initializing the auxiliary variable to zero; and

'8   assigning the auxiliary variable a non-zero value
9 upon execution of the basic block.

1   11. A method according to claim 9, wherein
2 instrumenting the SUT comprises:
3   determining a plurality of basic blocks comprised in
4 the SUT;
5   defining a single auxiliary variable for the SUT;
6   initializing the single auxiliary variable to zero;
7 and
8   assigning a unique non-zero value to the single
9 auxiliary variable upon execution of each basic block.

1   12. A method according to claim 1, wherein
2 computing the coverability metric comprises:
3   evaluating an attained coverability responsive to
4 the respective results produced by running the symbolic
5 model checker;
6   evaluating an unattained coverability responsive to
7 the respective results produced by running the symbolic
8 model checker;
9   performing a comparison between the attained
10 coverability and the coverability tasks;
11   calculating the coverability metric responsive to
12 the comparison; and
13   analyzing the behavioral model of the SUT with
14 respect to the unattained coverability.

1   13. A method according to claim 1, and comprising
2 analyzing a design of the SUT, responsive to the
3 coverability metric, for at least one of a group of
4 properties comprising dead code, unattainable states,
5 uncoverable statements, uncoverable states, unattainable
6 transitions, unattainable variable values, and
7 unreachable conditions.

1   14. A method according to claim 1, and comprising

`2 applying a testing strategy chosen from one of a group of
3 strategies comprising excluding uncoverable elements from
4 coverage measurements, setting coverage goals responsive
5 to the coverability metric, and determining a criterion
6 for stopping testing responsive to the coverability
7 metric.

1   15. A method according to claim 14, wherein the
2 uncoverable elements comprise one or more elements chosen
3 from a group of elements comprising uncoverable
4 statements, uncoverable states, unattainable transitions,
5 unattainable variable values, and unreachable conditions.

1   16. A method according to claim 1, wherein
2 formulating the respective coverability tasks for the
3 dominating blocks of the SUT comprises:
4   identifying a coverage model for the SUT;
5   defining a coverability model for the SUT responsive
6 to the coverage model; and
7   generating the respective coverability tasks
8 responsive to the coverability model.

1   17. A method for performing coverability analysis
2 in software, comprising:
3   formulating first and second coverability tasks for
4 software under test (SUT);
5   generating a rule regarding behavior of the SUT
6 corresponding to the first coverability task;
7   running a symbolic model checker comprising an
8 inflator to test a behavioral model of the SUT responsive
9 to the rule so as to produce an inflated result; and
10   evaluating the second coverability task responsive
11 to the inflated result.

1   18. A method according to claim 17, wherein
2 formulating the second coverability task comprises
3 choosing a plurality of coverability tasks from a set of

'4 all coverability tasks for the SUT, and wherein

5 evaluating the second coverability task comprises

6 evaluating the plurality.

1   19. A method according to claim 17, wherein

2 generating the rule regarding behavior of the SUT

3 comprises:

4   performing a static analysis of the SUT comprising:

5   identifying a set of dominating blocks in the SUT;

6 and

7   solving a subset cover problem on the set of

8 dominating blocks so as to produce a plurality of

9 dominating blocks; and

10   selecting the first coverability task responsive to

11 the plurality.

1   20. A method according to claim 19, wherein

2 selecting the first coverability task comprises:

3   identifying a greatest-influence dominating block

4 having a largest set of dominated blocks comprised in the

5 plurality; and

6   selecting the first coverability task responsive to

7 the greatest-influence dominating block.

1   21. A method according to claim 19, wherein the set

2 of dominating blocks comprises a set of all dominating

3 blocks in the SUT, and wherein the plurality of

4 dominating blocks comprises fewer blocks than the number

5 of all the dominating blocks.

1   22. A method according to claim 17, wherein running

2 the symbolic model checker comprises performing a number

3 of executions of the symbolic model checker, wherein the

4 number of executions is smaller than a total number of

5 coverability tasks for the SUT.

1   23. A method according to claim 17, and comprising

2 writing the SUT in a programming language adapted to

'3 define at least one of a group of elements comprising a

4 software element and a hardware element.

1   24. A method according to claim 17, wherein

2 formulating the first and second coverability tasks for

3 the SUT comprises formulating the tasks by at least one

4 of a group of methods comprising manual formulation and

5 automatic formulation.

1   25. A method according to claim 17, wherein

2 generating the rule regarding behavior of the SUT

3 comprises generating the rule by at least one of a group

4 of methods comprising manual generation and automatic

5 generation.

1   26. A method according to claim 17, wherein running

2 the symbolic model checker comprises evaluating the

3 inflated result and determining the truth or falsity of

4 the rule responsive to the evaluation.

1   27. A method according to claim 17, wherein

2 generating the rule comprises instrumenting the SUT by

3 adding one or more statements and one or more auxiliary

4 variables thereto, so as to facilitate evaluation of the

5 rule.

1   28. A method according to claim 27, wherein

2 instrumenting the SUT comprises:

3   determining a plurality of basic blocks comprised in

4 the SUT; and

5   for each basic block:

6   defining an auxiliary variable for the block;

7   initializing the auxiliary variable to zero; and

8   assigning the auxiliary variable a non-zero value

9 upon execution of the basic block.

1   29. A method according to claim 27, wherein

2 instrumenting the SUT comprises:

'3 '   determining a plurality of basic blocks comprised in

4   the SUT;

5   defining a single auxiliary variable for the SUT;

6   initializing the single auxiliary variable to zero;

7   and

8   assigning a unique non-zero value to the single

9   auxiliary variable upon execution of each basic block.

1   30.   A method according to claim 17, wherein running

2   the symbolic model checker comprises producing the

3   inflated result regardless of the truth or falsity of the

4   rule.

1   31.   A method according to claim 17, wherein

2   evaluating the second coverability task responsive to the

3   inflated result, comprises:

4   evaluating an attained coverability responsive to

5   the inflated result from running the symbolic model

6   checker;

7   evaluating an unattained coverability responsive to

8   the respective results produced by running the symbolic

9   model checker;

10   comparing the attained coverability with a plurality

11 of all coverability tasks for the SUT;

12   calculating a coverability metric responsive to the

13 comparison; and

14   analyzing the behavioral model of the SUT with

15 respect to the unattained coverability.

1   32.   A method according to claim 31, and comprising

2   analyzing a design of the SUT, responsive to the

3   coverability metric, for at least one of a group of

4   properties comprising dead code, unattainable states,

5   uncoverable statements, uncoverable states, unattainable

6   transitions, unattainable variable values, and

7   unreachable conditions.

1   33. A method according to claim 31, and comprising

2   applying a testing strategy chosen from one of a group of

3   strategies comprising excluding uncoverable elements from

4   coverage measurements, setting coverage goals responsive

5   to the coverability metric, and determining a criterion

6   for stopping testing responsive to the coverability

7   metric.

1   34. A method according to claim 33, wherein the

2   uncoverable elements comprise one or more elements chosen

3   from a group of elements comprising uncoverable

4   statements, uncoverable states, unattainable transitions,

5   unattainable variable values, and unreachable conditions.

1   35. A method according to claim 17, wherein running

2   the symbolic model checker comprises:

3   performing a plurality of executions of an inflator

4   program so as to produce a plurality of inflated results;

5   and

6   evaluating the second coverability task responsive

7   to the plurality of inflated results.

1   36. A method according to claim 17, wherein

2   formulating the first and second coverability tasks for

3   the SUT comprises:

4   identifying a coverage model for the SUT;

5   defining a coverability model for the SUT responsive

6   to the coverage model; and

7   generating the first and second coverability tasks

8   responsive to the coverability model.

1   37. Apparatus for performing coverability analysis

2   in software, comprising a computing system which is

3   adapted to perform a static analysis of software under

4   test (SUT) so as to identify a plurality of dominating

5   blocks in the SUT, formulate respective coverability

'6 tasks for the dominating blocks of the SUT, generate

7 rules regarding behavior of the SUT corresponding

8 respectively to the coverability tasks, run a symbolic

9 model checker to test a behavioral model of the SUT for

10 each of the rules so as to produce respective results for

11 the rules, and compute a coverability metric for the SUT

12 responsive to the results and the coverability tasks.

1 38. Apparatus for performing coverability analysis

2 in software, comprising a computer system which is

3 adapted to formulate first and second coverability tasks

4 for software under test (SUT), generate a rule regarding

5 behavior of the SUT corresponding to the first

6 coverability task, run a symbolic model checker

7 comprising an inflator to test a behavioral model of the

8 SUT responsive to the rule so as to produce an inflated

9 result, and evaluate the second coverability task

10 responsive to the inflated result.

1 39. A computer software product for performing

2 coverability analysis in software, comprising a

3 computer-readable medium having computer program

4 instructions recorded therein, which instructions, when

5 read by a computer, cause the computer to perform a

6 static analysis of software under test (SUT) so as to

7 identify a plurality of dominating blocks in the SUT,

8 formulate respective coverability tasks for the

9 dominating blocks in the SUT, generate rules regarding

10 behavior of the SUT corresponding respectively to the

11 coverability tasks, run a symbolic model checker to test

12 a behavioral model of the SUT for each rule so as to

13 produce respective results for the rules, and compute a

14 coverability metric responsive to the results and the

15 coverability tasks.

1 40. A computer software product for performing

2   coverability analysis in software, comprising a

3   computer-readable medium having computer program

4   instructions recorded therein, which instructions, when

5   read by a computer, cause the computer to formulate first

6   and second coverability tasks for software under test

7   (SUT), generate a rule regarding behavior of the SUT

8   corresponding to the first coverability task, run a

9   symbolic model checker comprising an inflator to test a

10 behavioral model of the SUT responsive to the rule so as

11 to produce an inflated result, and evaluate the second

12 coverability task responsive to the inflated result.